WHAT IS CLAIMED IS:

1. A system for router packet control and ordering comprising:

a packet forwarding engine containing a plurality of fast data paths each interconnected with an input port and an output port;

a processing block interconnected with said input ports of said packet forwarding engine through a plurality of rate matching queues;

an ingress ASIC containing a packet ordering block, said packet ordering block interconnected with said output ports of said packet forwarding engine through a plurality of reorder queues within said ingress ASIC; and

a packet data RAM memory interconnected with said ingress ASIC.

2. The system of claim 1 wherein said packet forwarding engine contains an exception processor, such that said fast data path bypasses said exception processor and such that said fast data path is interconnected with said exception processor through an exception data path.

3. The system of claim 1 wherein said input ports of said packet forwarding engine include a wide bandwidth port and a narrow bandwidth port.

4. The system of claim 3 wherein said wide bandwidth port has a capacity of approximately 3.2 gigabits per second (Gb/s).

5. The system of claim 3 wherein said narrow bandwidth port has a capacity of approximately 1.6 Gb/s.

6. The system of claim 1 wherein said reorder queues include an exception reorder queue.

7. The system of claim 1 wherein said packet ordering block contains a now-serving counter.

8. The system of claim 1 wherein said processing block is interconnected with an input data link having a bandwidth capacity of approximately 10 Gb/s.

9. The system of claim 1 wherein said processing block is interconnected with multiple input data links, such that each said data link has a bandwidth capacity of at least 1.0 Gb/s.

10. The system of claim 9 wherein each said data link has a bandwidth capacity of approximately 2.5 Gb/s.

11. The system of claim 1 comprising a plurality of said packet forwarding engines, such that each said packet forwarding engine contains a plurality of fast data paths each interconnected with an input port and an output port, each said input port interconnected with said processing block through a plurality of rate matching queues; and each said output port interconnected with said packet ordering block through a plurality of reorder queues within said ingress ASIC.

12. The system of claim 11 wherein said processing block is interconnected with an input data link having a bandwidth capacity of approximately 10 Gb/s.

13. The system of claim 11 wherein said processing block is interconnected with multiple input data links, such that each said data link has a bandwidth capacity of at least 1.0 Gb/s.

14. The system of claim 13 wherein each said data link has a bandwidth capacity of approximately 2.5 Gb/s.

15. The system of claim 11 wherein said processing block is partitioned into a plurality of independent processing sub-blocks.

16. The system of claim 11 wherein said packet ordering block is partitioned into a plurality of independent packet ordering sub-blocks.

17. A method of control and ordering of information packets in a network router comprising:

receiving through an input data link said packets;

sequence numbering said packets;

distributing said packets among a plurality of rate matching queues, such that each said rate matching queue feeds one of substantially parallel data links having differing information rates;

flowing said packets through said plurality of substantially parallel data links at differing information rates; and

reordering said packets after said flowing.

18. The method of claim 17 wherein said substantially parallel data links are limited to a maximum packet flow rate of approximately 2.1 million packets per second.

19. The method of claim 17 wherein a portion of said substantially parallel data links are narrow bandwidth data links.

20. The method of claim 19 wherein a portion of said substantially parallel data links are wide bandwidth data links, such that the maximum bandwidth limit of said wide bandwidth data links is greater than the maximum bandwidth limit of said narrow data links.

21. The method of claim 20 wherein the maximum bandwidth of said narrow bandwidth data links is approximately 1.6 gigabits per second (Gb/s).

22. The method of claim 20 wherein the maximum bandwidth of said wide bandwidth data links is approximately 3.2 Gb/s.

23. The method of claim 17 wherein said packets are received through said input data link at a rate of approximately 10 Gb/s.

24. The method of claim 17 further comprising processing said packets within each of said plurality of substantially parallel data links, such that said processing is limited to a maximum data rate of approximately 2.5 Gb/s.

25. The method of claim 20 wherein said packets are distributed among said rate matching queues according to packet size, such that large packets greater than a threshold size are placed in large packet queues feeding wide bandwidth data links and such that small packets not greater than said threshold size are placed in small packet queues feeding narrow bandwidth data links.

26. The method of claim 25 wherein said small packet is placed in a particular small packet queue having the greatest unoccupied space.

27. The method of claim 25 wherein said large packet is placed in a particular large packet queue having the greatest unoccupied space.

28. The method of claim 25 wherein said packets are distributed roughly equally among said rate matching queues using a round robin algorithm.

29. The method of claim 25 wherein said threshold size is approximately 200 bytes.

30. The method of claim 25 further comprising inserting a keep-alive packet into any narrow bandwidth data link that is idle for a predetermined period of time.

31. The method of claim 30 wherein said predetermined period of time is programmable.

32. The method of claim 30 wherein said predetermined period of time is on the order of 10 microseconds.

33. The method of claim 17 wherein said sequence numbering is applied within a header prepended to said packet.

34. The method of claim 17 wherein said packets comprise fast path packets and exception packets different from said fast path packets.

35. The method of claim 34 wherein said exception packets are distinguished from said fast path packets by setting a bit within a header prepended to each said packet.

36. The method of claim 34 wherein said exception packets are processed within a portion of said plurality of substantially parallel data links that is bypassed by said fast path packets.

37. The method of claim 36 wherein said exception packets are reordered separately and independently from said fast path packets.

38. The method of claim 37 wherein said reordering of fast path packets comprises:

receiving said fast path packets from said substantially parallel data links;

separating said packet headers from said packet payloads;

loading said packet headers into a plurality of fast path reorder queues; and

removing said packet headers from said fast path reorder queues in order of said sequence numbering.

39. The method of claim 38 wherein said removing comprises:

a. incrementing a now-serving number by one unit;

b. if a packet header having a sequence number matching said now-serving number is in said plurality of fast path reorder queues, then removing said packet header and iterating steps 39-a and 39-b; otherwise

c. after a subsequent packet header arrives in each of said fast path reorder queues, if a packet header having a sequence number matching said now-serving number is in said plurality of fast path reorder queues, then removing said packet header and iterating steps 39-a, 39-b and 39-c; otherwise

d. iterating steps 39-a, 39-b, and 39-c until a packet header is found having a sequence number matching said now-serving number.

40 The method of claim 39 wherein, if any of said plurality of fast path reorder queues does not receive a subsequent packet header during a programmable elapsed period of time, then inserting a keep-alive packet into a fast path reorder queue.

41. The method of claim 40 wherein said programmable elapsed period of time is on the order of 10 microseconds.

42. The method of claim 38 further comprising storing said separated packet payloads in a memory device.

43. The method of claim 37 wherein said separate and independent reordering of exception packets comprises:

receiving said exception packets from said substantially parallel data links;

separating said exception packet headers from said exception packet payloads;

loading said exception packet headers into a plurality of exception reorder queues; and

removing said exception packet headers from said exception reorder queues in order of said sequence numbering.

44. The method of claim 43 further comprising storing said separated exception packet payloads in a memory device.

45. The method of claim 43 wherein said removing comprises:

a. if an exception packet header exists in each of said plurality of exception reorder queues, then removing the exception packet header having the lowest sequence number; otherwise

b. after a time-out mechanism occurs, then removing the exception packet header having the lowest sequence number among all exception packets in any of said exception reorder queues.

46. The method of claim 45 wherein said time-out mechanism is programmable.

47. The method of claim 43 wherein nonordering packets are created within said portion of said plurality of substantially parallel data links wherein said exception packets are processed, said nonordering packets being identified by a bit set within a header of said nonordering packets, such that a said identified nonordering packet is loaded into one of said reorder queues and upon reaching the head of said reorder queue is removed from said reorder queue without regard to any sequence numbering.